Creative Assessment in Programming: Diversity and Divergence

Steven Bradley
s.p.bradley@durham.ac.uk
Dept Computer Science, Durham University
Durham, UK

ABSTRACT

Negative stereotypes persist in computing, and align poorly with research into the motivations of female students. In particular, female students are more inclined to want to work creatively and have a positive impact through their work. However programming assignments are often tightly constrained and rather pointless in themselves so are doubly unattractive. Alongside this, concerns are often raised about plagiarism in programming assignments, particularly when the assessment process is automated. We attempt to address both of these issues by designing more creative programming assignments, allowing students to engage in work aligned with whatever their interests are. By providing a more divergent assessment, automated plagiarism detectors are much more effective because the likelihood of false positives is much lower than in more constrained, convergent assessments. We also show how to combine this with partial automation of assessment. To examine this approach we compare the results of two subsequent years of delivery of the same second-year undergraduate programming module, and find that, using more creative assessments, female students average scores were substantially increased so that they outperform male students. While the results are not quite statistically significant (according to 2-way ANOVA), they demonstrate potential that could be verified with a larger sample.

CCS CONCEPTS

• **Social and professional topics** → **Student assessment**; • **Applied computing** → *Computer-assisted instruction.*

KEYWORDS

programming, assessment, diversity, plagiarism, automation

1 INTRODUCTION: WHY?

By using creative assignments we seek to address two separate issues: firstly the lack of gender diversity within computing; secondly the problem of detecting plagiarism in programming assignments.

Gender

We all know that there are not enough girls and women in computing, which is disadvantageous both to women and to computing. In the UK in 2014 the proportion of girls choosing to study GCSE computing (14-16) was only 15%[4] , and similarly low (17.6%) on entry to university in 2017 [12].

This trend appears start between the ages of 11 and 15 [18] and although there is no one simple answer, a large part of the problem appears to be in the existence and perpetuation of negative stereotypes of the culture [5] such as

- socially isolated
- not collaborative
- masculine interests

The perceived social good of computing [10] is also of more importance to female students than males. According to Benyo and White [1], who studied the opinions of 13-17 year-olds in the USA, 56% of girls said that "having the power to do good and doing work that makes a difference" was extremely important, as opposed to only 46% of boys. Similarly, 39% of girls (and 35% of boys) felt that "having an opportunity to express yourself creatively" was important. "Earning a high salary" was of equal importance to creativity for girls (39%) as opposed to boys, for whom it was much higher at 50%. Another study [9] of 12-year-olds in Norway reported that "creativity is an excellent means to promote and teach programming, and ... a workshop approach raises interest in computer science among female students in particular."

Plagiarism

In computer science education plagiarism has long been recognised as a problem, with programming being particulary problematic [15, 16]. One study [6], found 33% of students reporting that they had plagiarised and 30% said they had knowingly been plagiarised. In the same study only 64% of staff had identified a case of plagiarism and 12% of the staff had never heard of a case of plagiarism, suggesting that a lot of student plagiarism is not identified by staff. This exposes a fundamental tension between reliability and validity of assessment [17]. Many computing academics identify programming coursework as a more valid way of assessing students than a written exam, but in a study by Sheard et al. [21] one of the academics reported

> "We do exams because of plagiarism, there is no other reason for doing an exam"

So plagiarism, which is a profound threat to reliability, tends to push academics towards exams, which are a threat to validity. To address this there are many plagiarism detection tools for programming such as MOSS [2] and JPlag [19]. Hage, Rademaker and Vugt [11] identified 18 separate plagiarism detection projects/tools in 2010, and more have been developed since [7, 8]. One of the issues that arises with these tools is that false positives make it difficult to identify plagiarism accurately, when different students may arrive at the same solution independently i.e. the assessment is convergent (as opposed to divergent) [17].

Having divergent assessment is a challenge when used in parallel with automated assessment, often adopted to reduce feedback times, although techniques such as randomisation of assessment can allow automation of divergent assessment [3]. In this paper we discuss another approach, which is to allow creativity in summatively assessed assignments, hopefully making it more interesting to female students, while increasing the divergence of assessment.

2 CREATIVE ASSESSMENT IN PROGRAMMING: WHAT?

The basic idea is to provide much more student choice is programming assignments, requiring them to choose a domain of application in which to demonstrate the skills and techniques that are to be assessed. Before making this change we would identify the set of practices to be assessed (e.g. writing and using a REST API) and then construct a scenario in which this could be demonstrated (e.g. build a web-site for managing events). So before making the change the high level summary of the assessment task was

- People attend events at venues at certain times. You need to write a web app to allow a list of events to be searched and updated
- Provide a responsive single page web app for events
- Provide a nodejs web service through a REST API
- A separate service provides authentication (part of a microservices architecture)
- Server to be deployed both locally (for testing) and in cloud
- Integrate an external web service to provide more event data

In the new style the task is very similar, but less constrained:

- Construct a dynamic web-site for a domain of your choosing, as long as it includes people and another item type as well as people e.g. events, photos, comments
- Provide a responsive single page web app
- Provide a nodejs web service through a REST API
- Server to be deployed both locally (for testing) and in cloud

Some specific details (e.g. authentication and access to external API) were not specified in the new-style assignment. Both assignments were assessed by a combination of automated assessment and faculty review of a two-minute video that students produced to demonstrate their program. To do the automated assessment in the 'creative' case, the part of the REST API that dealt with people was specified, but the rest of the API was left unspecified. The test cases for the 'people' part of the API were provided to students, with care to ensure that the test would still be passed if the data fields associated with people were extended beyond the basic requirement of the assignment (username, forename, surname).

For the first assignment, the API was fully specified through examples such as (slightly modified)

- For `GET BASE/venues`
- List all the venue details
- No parameters
- Example response. Your response must follow this structure exactly, but with your data in it

```
{
"venues":{
"v_1":{
"name":"First venue",
"postcode":"AB12 3CD",
"town":"Placeville",
"url":"http://www.wherever.com",
"icon":"http://www.wherever.com/icon.png"
}
}
```

Examples of responses were provided as a Postman [14] collection.

For the second assignment the API was partly specified through descriptions e.g.

- For `GET /people`
- Before any people or relationships added

```
[{"username":"doctorwhocomposer",
 "forename":"Delia",
 "surname":"Derbyshire"}]
```

Jest [13] test cases were also provided e.g.

```
test(
'GET /people includes doctorwhocomposer',
() => {
    return request(app)
        .get('/people')
        .expect(/doctorwhocomposer/);
});
```

Aside from the differences to the assignment specifications and the contextual differences (see section 2), there was one other difference in the teaching of the module, which we describe here, partly to make the reader aware of the pedagogical context so as to judge the impact of the change, and partly because it is slightly novel in itself.

Managing sequence with trello

One of the challenges of teaching web programming (and indeed any other kind of programming in higher education) is that the students exhibit a wide variety in their previous exposure to the content. In teaching the second cohort, one aspect of agile teaching [20] was adopted to select and sequence the material that was presented in lectures and set in practicals. We used trello, a kanban-style list/task tracker, putting topics into headings of

- To do
- This lecture
- Next practical
- Next lecture
- Done

To do this the students were presented with the assignment early in the course (lecture 2/10), i.e. before most of the relevant material had been covered. The topics in the assignment were then identified in-class by the students and topics for the next lecture and next practical selected by in-class voting. This procedure was repeated at the end of each lecture, treating the week's learning as an agile sprint. Ideally we would have had formative tests to assess whether or not the topic was satisfactorily covered, but that was a step too far in terms of preparation and class time.

Some topics ended up not being covered directly in a session because the students felt that lecture time would be more usefully spent giving more detail on some topics, rather than on other topics which could be accessed through on-line resources.

Figure 1 shows the state of the trello board at the end of the lecture course. In practice the sequencing of the material was very

similar to the lecturer-sequenced content from the first cohort, except that

- Server-side JavaScript (nodejs) was covered before client-side JavaScript (DOM manipulation)
- More time was spent in demonstrating the development of a REST/AJAX example
- More of the material was left for individual study by the students

Context: Where?

The study took place in a small UK university (14,000 undergraduate students) with entry requirements for computer science placing it in the top ten universities in the UK. Two web programming assignments in consecutive academic years of a second-year programming module were used. Web programming formed a quarter of a single compulsory 20 credit module and was taught by 10 one-hour lectures and 5 two-hour practical sessions. In both cases web programming was summatively assessed by a single programming exercise, as described in this paper. Cohort A (before the change) consisted of 89 students and cohort B (after the change) had 113 students. In both cases most of the students taking the module were studying single-honours computer science, with the others taking joint honours including at least a third of their second-year modules in computer science. In cohort A 10 of the students (11%) identified as female and in cohort B 20 students (18%) identified as female.

3 EVALUATION: DOES IT WORK?

Diversity

To evaluate how effective the approach was, we first of all look at the mean mark achieved by male and female students over the two cohorts and see how they compare, as shown in the table below.

Cohort/Gender	Female	Male	Total
A	62.3	70.3	69.4
B	73.9	71.8	72.1
Total	70.0	71.1	70.9

From this we can see the mean scores achieved by the two cohorts overall are very similar (A has 69.4 and N has 72.1), and overall the genders have very similar scores (70.0 for female and 71.1 for male). Also male students performed very similarly in the two cohorts (70.3 and 71.8) but the female students performed much better in cohort B, where the creative assessment was used, with the mean mark increasing by more than 10% from 62.3 to 73.9. This is a very pleasing result, but we need to check whether it is statistically significant, given the relatively small cohort size.

Running a 2-ay ANOVA test on the data, we assess the the influence of two categorical variables (cohort and gender) on a continuous dependent variable (mark), showing the results in the next table.

	F	PR(>F)
C(Cohort)	1.714331	0.191942
C(Gender)	0.213609	0.644461
C(Cohort):C(Gender)	2.542736	0.112398

The most important figure to note here is that $PR(> F)$ for C(Cohort):C(Gender) is 0.11. For a statistically significant result (95%) this value should be less than 0.05. So while the result is not significant at that level, the test indicates that there is only a 1 in 9 chance of the improved performance of female students in cohort B happening randomly. To get a more significant result we would need a larger effect size (although 11% is already large), a paired population study (difficult to control for the assignment, as it can't be repeated) or a larger cohort size. Even without statistical significance this result is promising.

The creativity in the approach was also supported in module evaluation questionnaires by students:

> "Enjoyed the freedom given in the web assignment, so could tailor to your interests."

> "Was happy that ... let us be creative with the assignment made it really fun and was a good idea."

but there was (only) one negative comment about the flexibility afforded to students

> "The coursework assignment made it unclear the detail to which certain features needed to be implemented"

Also, student input into sequencing was appreciated by some

> "being able to choose what was taught helped us with the coursework"

but not others

> "I think the lectures should be pre-structured as opposed to being dynamically chosen"

Divergence

A large range of topics were covered by the submissions, ranging from sports clubs to makeup, from photography to poetry and from vampires to the Avengers. There was no evidence of plagiarism in the submissions, but given the divergence of the submitted work it would be very easy to support a plagiarism case if it did occur, as there is a vanishingly small likelihood of two submissions being the same by chance.

4 CONCLUSIONS

While this was not set up as a research experiment, there is good evidence that female students did better in the creative assignment, encouraging diversity within the computing cohort. There are other factors to be taken into account between the cohorts that may also have had an effect:

- There were slightly different tasks and weightings between the assignments for the two cohorts
- Practical classes were not compulsory for the second cohort, and hence not as well attended. This may have favoured more diligent students.

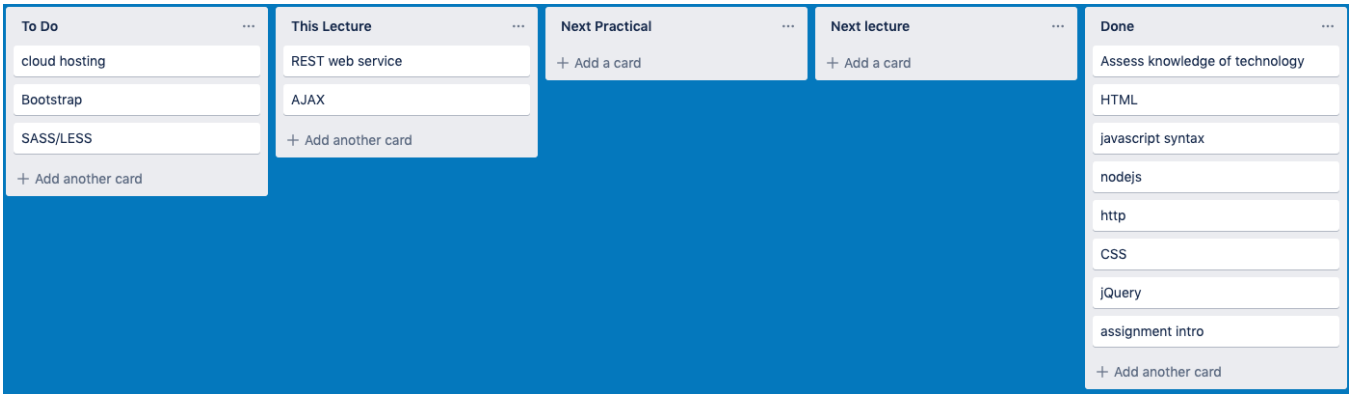

Figure 1: Trello board at the end of the lecture course

- There were twice as many females, and a significantly higher proportion, in the second cohort. This may have adversely affected performance of female students in the first cohort
- The second cohort influenced the sequencing of material

We adopted a similar approach in our first-year programming module, but this was alongside some other significant changes (including change of programming language) so there was no sound basis for comparison. However, in the module evaluation questionnaires for that module there was also good student support for the creativity introduced into the assignments.

Ideally in the future we would like to look at the impact of this approach in other modules in other universities, and to assess formally the change in divergence of submissions.

REFERENCES

[1] Julie Benyo and John White. 2009. *New Image for Computing Report on Market Research.* Technical Report. WGBH/ACM. http://www.dotdiva.org/downloads/New_Image_Computing_final.pdf

[2] K. W. Bowyer and L. O. Hall. 1999. Experience using "MOSS" to detect cheating on programming assignments. In *Frontiers in Education Conference, 1999. FIE '99. 29th Annual*, Vol. 3. 13B3/18–13B3/22 vol.3. https://doi.org/10.1109/FIE.1999.840376

[3] Steven Bradley. 2016. Managing Plagiarism in Programming Assignments with Blended Assessment and Randomisation. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research (Koli Calling '16)*. ACM, New York, NY, USA, 21–30. https://doi.org/10.1145/2999541.2999560

[4] T Bramley, C.L. Vidal Rodeiro, and S Vitello. 2015. *Gender differences in GCSE.* Cambridge Assessment Research Report. Cambridge Assessment. http://www.cambridgeassessment.org.uk/Images/gender-differences-at-gcse-report.pdf

[5] Sapna Cheryan, Allison Master, and Andrew N. Meltzoff. 2015. Cultural stereotypes as gatekeepers: increasing girls' interest in computer science and engineering by diversifying stereotypes. *Developmental Psychology* 6 (2015), 49.

[6] D. Chuda, P. Navrat, B. Kovacova, and P. Humay. 2012. The Issue of (Software) Plagiarism: A Student View. *IEEE Transactions on Education* 55, 1 (Feb. 2012), 22–28. https://doi.org/10.1109/TE.2011.2112768

[7] G. Cosma and M. Joy. 2012. An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis. *IEEE Trans. Comput.* 61, 3 (March 2012), 379–394. https://doi.org/10.1109/TC.2011.223

[8] Christian Domin, Henning Pohl, and Markus Krause. 2016. Improving Plagiarism Detection in Coding Assignments by Dynamic Removal of Common Ground. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16)*. ACM, New York, NY, USA, 1173–1179. https://doi.org/10.1145/2851581.2892512

[9] Michail N. Giannakos, Letizia Jaccheri, and Roberta Proto. 2013. Teaching Computer Science to Young Children Through Creativity: Lessons Learned from the Case of Norway. In *Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research (CSERC '13)*. 10:103–10:111.

[10] Michael Goldweber, John Barr, Tony Clear, Renzo Davoli, Samuel Mann, Elizabeth Patitsas, and Scott Portnoff. 2013. A Framework for Enhancing the Social Good in Computing Education: A Values Approach. *ACM Inroads* 4, 1 (March 2013), 58–79.

[11] Jurriaan Hage, Peter Rademaker, and Nike van Vugt. 2010. *A comparison of plagiarism detection tools.* Technical Report UU-CS-2010-015. Department of Information and Computing Sciences, Utrecht University. http://www.cs.uu.nl/research/techreps/repo/CS-2010/2010-015.pdf

[12] HESA. 2019. What do HE students study? | HESA. https://www.hesa.ac.uk/data-and-analysis/students/what-study

[13] Facebook Inc. [n.d.]. Jest Âů ðŞČŘ Delightful JavaScript Testing. https://jestjs.io/

[14] Postman Inc. [n.d.]. Postman | The Collaboration Platform for API Development. https://www.getpostman.com

[15] Alastair Irons. 2004. Reducing plagiarism in computing. In *Effective learning and teaching in computing*, Alastair Irons and Sylvia Alexander (Eds.). Routledge-Falmer, London, 100–110.

[16] M. Joy and M. Luck. 1999. Plagiarism in programming assignments. *IEEE Transactions on Education* 42, 2 (May 1999), 129–133. https://doi.org/10.1109/13.762946

[17] Mhairi McAlpine. 2002. *Principles of assessment.* CAA Centre, University of Luton.

[18] Research Microsoft. 2017. Why don't European girls like science or technology? https://news.microsoft.com/europe/features/dont-european-girls-like-science-technology/

[19] Lutz Prechelt and Guido Malpohl. 2002. Finding Plagiarisms among a Set of Programs with JPlag. *Journal of Universal Computer Science* 8, 11 (2002). https://doi.org/10.3217/jucs-008-11-1016

[20] Jason H. Sharp and Guido Lang. 2018. Agile in Teaching and Learning: Conceptual Framework and Research Agenda. *Journal of Information Systems Education* 29, 2 (May 2018), 45. https://jise.org/Volume29/n2/JISEv29n2p45.html

[21] Judy Sheard, Simon, Angela Carbone, Daryl D'Souza, and Margaret Hamilton. 2013. Assessment of Programming: Pedagogical Foundations of Exams. In *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '13)*. ACM, New York, NY, USA, 141–146. https://doi.org/10.1145/2462476.2465586